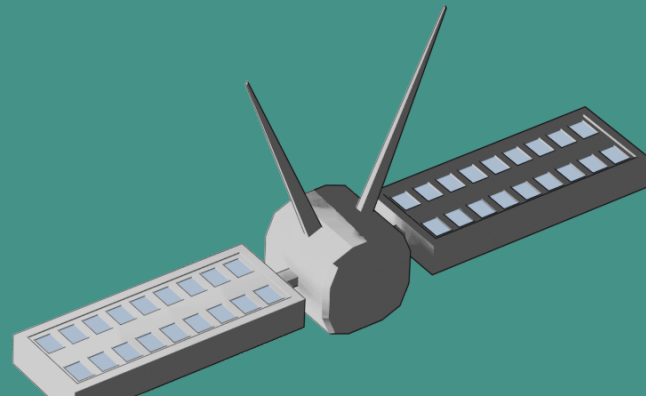
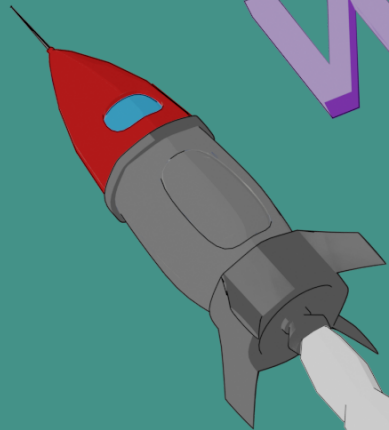
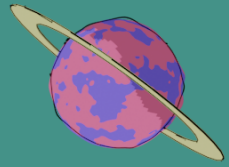


Indico 3.5 Workshop



Looking into the future

Roadmap • Wishlist • Ideas

Adrian Mönnich • Indico Workshop 3.5 • March 2023
Picture: "Crystal Ball" by Jeffrey (CC BY-ND)







v3.3

- **Privacy features** Part 2
 - Yep, we still love GDPR (...and CERN's own GDPR aka "OC-11")
 - User-accessible dump (JSON?) of everything they are linked to
 - User consent for privacy notice during registration
 - Admin tool to anonymize an account (deletion not always feasible)
- Receipt/certificate generation
 - Fully customizable templates for PDF generation

v3.4

- **Rewrite conference timetable** & timetable management in React
 - Rare cases will take the most time... e.g. heavily parallel conferences
 - Maybe **refresh conference layout** to have more horizontal space?
- Remove most (all?) ultra-legacy JS (from before jQuery)
 - Opens the door for improving our asset building/bundling pipeline

vX.Y - "Wishlist"

- **User-focused home page** 
 - More dashboard-like
 - Take user's favorites  into account
- **UX/UI improvements**
 - More consistency
 - Accessibility 
 - Material editor: Drag & Drop
 - Better support for small screens 
- **Modern API**
 - Legacy ("classic") API is from 2011 and feels that way
 - API keys already deprecated in favor of Bearer tokens
 - Granular scopes? Versioned endpoints? OpenAPI specs?
 - Modify/create things via API as well?

Does our versioning (still) make sense?

- v1 to v2: ZODB -> Postgres, **nearly full rewrite**
- v2 to v3: Python 2 -> Python 3, **great for devs & future-proof**
- v3 to v4: ???

"Minor" releases are actually major features & often DB changes

"Patch" releases are normal releases with features and bugfixes

Ditch one version digit?

- All the major web browser vendors do it
 - But they also have a pretty static release cadence (~monthly)
 - 111.0, 111.1, 111.2, 112.0, 113.0, ...
 - Are "minor" releases supposed to include (small) features?
- We do not need a bugfix-only branch
 - Limits dev flexibility & adds overhead (branch switching & merging)
 - Almost continuous deployment (at CERN); small features are not held back for weeks

How it could look like for us

- v3.3 becomes v4.0 instead (and v3.4 would be v5.0)
- Once v4.0 is released, that branch will become v4.1-dev for bugfixes and smaller "straight to prod" features
- Heavy new development would then target v5.0

But is it really worth it?

- It's just a number!
- Nobody expects *semver* outside libraries
 - API versioning is a completely different topic!
 - We never got complaints about our versioning/release schema

 If you have any thoughts or opinions on this topic let me know!

indico